2022 IEEE International Conference on
Autonomous Robot Systems and Competitions (ICARSC)
Santa Maria da Feira, Portugal – April 29-30, 2022

2022 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC) | 978-1-6654-8217-2/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICARSC55462.2022.9784771

# Real-time Vision Based Obstacle Detection in Maritime Environments

Duarte Nunes*, João Fortuna*, Bruno Damas†‡, and Rodrigo Ventura‡
*Ocean & Space Unit - Department of Electronics & Embedded Systems
CEiiA - Centro de Engenharia e Desenvolvimento, Matosinhos, Portugal
Email: {duarte.nunes, joao.araujo}@ceiia.com
†CINAV - Centro de Investigação Naval, Almada, Portugal
‡Institute for Systems and Robotics, Instituto Superior Técnico, Lisbon, Portugal
Email: {bdamas, rodrigo.ventura}@isr.tecnico.ulisboa.pt

***Abstract*—Automatic obstacle detection is a key feature for unmanned surface vehicles (USV) operating in a fully autonomous manner. While there are currently many approaches to obstacle detection in maritime environments (*e.g.*, LiDAR, radar) the proposed approach resorts to standard, inexpensive RGB cameras to perform the detection of such obstacles. Recent advances in deep neural network detectors achieve state-of-the-art detection results, and some one-stage networks achieve very good results while maintaining inference times small enough to be compatible with real-time capabilities on low-cost embedded processing units.**

**In this paper, we train the YOLO v4 network to detect different types of ships, using publicly available maritime datasets. After training, we evaluate the obtained network on the processing unit located onboard the UAV with respect to detection accuracy and real-time processing capability, thus demonstrating that the presented detection method can be considered a robust, fast, flexible, and inexpensive approach to obstacle detection in USV applications.**

***Index Terms*—Deep Learning, Convolutional Neural Networks, Object Detection, Embedded Systems, Unmanned Surface Vehicles**

## I. Introduction

Unmanned Surface Vehicles (USVs) are increasingly being used for military purposes, security applications such as harbour and coastline patrol, environmental monitoring, bathymetric mapping and robotic research applications [1]. Moreover, USVs can be deployed in dangerous environments like mine countermeasures operations and hazard areas contaminated by biological or chemical substances, without endangering navy personnel. These vehicles offer a greater autonomy and can carry a larger payload, when compared to aerial or underwater vehicles, and can be used as communication relays between heterogeneous vehicles during coordinated operations, providing a link between acoustic communications with underwater vehicles and standard radio-frequency communications with aerial, ground, and other surface units.

Construction and development of USVs started to draw a substantial attention in the beginning of this century, with MIT's catamaran ACES [2] and kayak SCOUT [3], demonstrating the autonomous navigation and control capabilities of such vehicles and the possibility of acquiring hydrographic data autonomously. At the same time other civilian USVs were being developed in other research laboratories around the globe, such as Portuguese catamaran Delfim (Lisbon ISR/IST) [4] and catamaran Roaz (Porto LSA/ISEP) [5], providing a testbed for cooperation between autonomous vehicles in maritime operations and data acquisition. This kind of vehicles usually follow a modular construction approach to provide an increased flexibility for research purposes and easy adaptation for different types of payloads and missions.

USVs are also operated by many navies in the context of military and security applications. With very different sizes and payloads, they range from the small Israeli Stingray USV[1], built for port security and harbour defence missions, and the well known Sea Hunter[2], a $40\,\mathrm{m}$ USV operated by the US navy, going through other types of USVs like the Protector[3] (Israel), the ULAQ[4] (Turkey) and the JARI[5] USV (China), to name just a few.

While military USVs are typically remotely controlled by a human operator, the trend in civil applications is shifting towards fully autonomous operation. This requires, in addition to autonomous guidance and navigation algorithms, robust obstacle avoidance methods relying on the vehicle sensors [6]–[8]. Sensors commonly used for obstacle detection in maritime environments include active-ranging approaches employing sonar, radar and/or LiDAR [8]. Sonar and radar provide a high depth resolution and accuracy, together with a good near-range (LiDAR) or long-range (radar) obstacle detection capabilities. However, using this type of sensors to provide information regarding the type of detected obstacle can be difficult without complementary information. This is a serious limitation if the USV is supposed to operate near other human operated boats and ships, as the International Regulations for Preventing Collisions at Sea (COLREGS) provides a set of

This work was supported by the LARSyS - FCT Project UIDB/50009/2020

[1] https://defense-update.com/20061121_stingray.html
[2] https://vigor.net/projects/sea-hunter
[3] https://www.rafael.co.il/worlds/naval/usvs/
[4] https://www.ulaq.global/
[5] https://www.china-arms.com/jari-usv-first-trial

978-1-6654-8217-2/22/$31.00 ©2022 IEEE

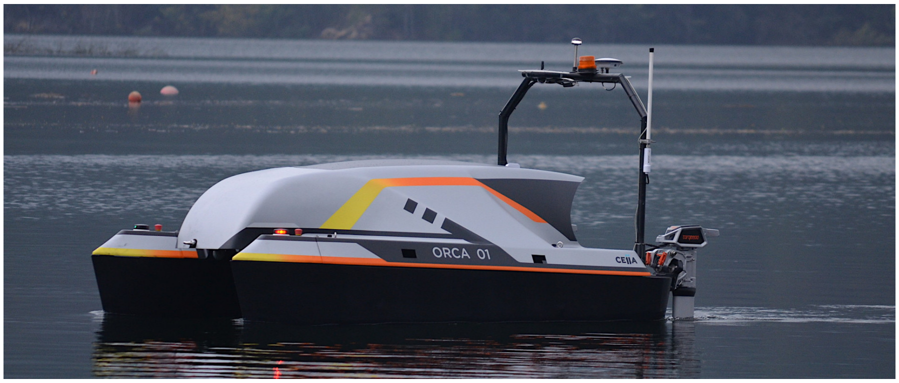

Fig. 1. CEiiA's ORCA USV.

rules that specify how each vessel should behave and which vessel should give way in several circumstances like crossing, overtaking or head-on situations [9], [10]. These rules depend on the vessel type: sailing vessels, for instance, are limited as to their manoeuvrability in the presence of forward wind or in the absence of wind. Low cost optical sensors like electro-optical and IR sensors, on the other hand, can identify the type of obstacle when coupled with vision based object detection algorithms trained on the type of data retrieved by the sensor, for example an RGB image. With this coupling they are able to discriminate for instance between ships, sail boats and buoys, and can complement or replace the aforementioned active-ranging sensors. Additionally, they typically have lower acquisition costs and lower power consumption.

Recent advances in deep neural network based approaches for object detection in RGB images have achieved state-of-the-art detection performance while, at the same time, keeping inference times low enough to allow for real-time obstacle avoidance capabilities in low-cost, commercial off-the-shelf embedded processing units. In this paper, we train the state-of-the-art fast detection YOLO v4 network to detect different types of ships, using publicly available maritime datasets. After training, this network runs in a Jetson Nano processing unit on board the ORCA vehicle, an USV developed by CEiiA for bathymetric mapping, seabed monitoring and other scientific applications. The ORCA USV[6], depicted in Fig. 1, is a differential-drive electric catamaran. Dimensions are $3.4\,\mathrm{m}$ overall length and $1.7\,\mathrm{m}$ beam, weighing $450\,\mathrm{kg}$. We show that using such detector is a viable approach for object detection and classification in maritime scenarios, enabling real-time obstacle detection resorting to standard hardware and RGB cameras. As ORCA is intended to operate in crowded environments in a fully autonomous way, such object detection is crucial for complying to COLREGS in such situations (Fig. 2).

The paper is organized as follows: in Section II we provide some background on deep learning detection algorithms and vision based obstacle detection in maritime environments; in Section III we present the details on the approach taken and on the datasets used; after that we provide and discuss some experimental results in Section IV and finally, in Section V we provide some concluding remarks and suggestions for future work.

Fig. 2. ORCA USV operating in a harbour.

## II. Background and Related Work

State-of-the-art solutions to object detection in maritime environments using standard RGB cameras explore either image segmentation techniques that extract objects from the background, or the capabilities of detection networks to identify and locate objects directly in an image.

The WaSR algorithm [11] shows state-of-the-art image segmentation performance, introducing new encoder and decoder paths to perform water segmentation, augmented by an IMU unit to correct segmentation errors. The use of IMU units in WaSR takes inspiration from [12] and [13], both papers demonstrating the use of IMU data in correcting the perturbations experienced by the surface vehicle that propagate to the onboard cameras. WODIS [14] also uses

[6] https://orca.ceiia.com/

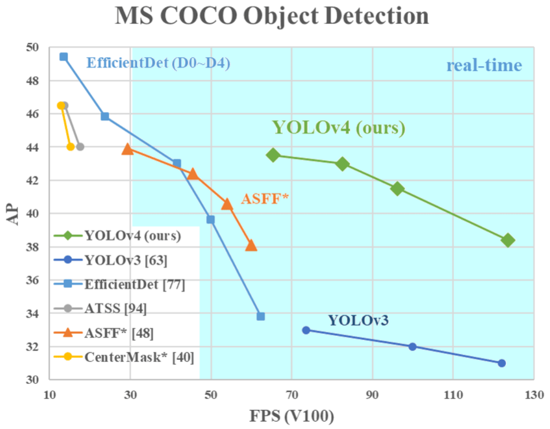

Fig. 3. Comparison of state-of-the-art detectors with respect to detection performance (AP) and speed (FPS) in the MS COCO dataset. Image taken from [21].

semantic segmentation to separate maritime obstacles from the ocean/sky backgrounds. Despite showing promising results, semantic segmentation algorithms are notoriously computationally demanding and are not currently compatible with real-time operation on standard embedded hardware.

Current state-of-the-art detection neural networks are more versatile with respect to the required computational power, and some variants of these networks were developed with faster inference and real-time detection capabilities in mind. While most of these networks perform detection on standard RGB images, some works perform object detection with already popular or commonly implemented sensors: in [15] a multi-modal sensor fusion framework is presented that enriches object detection with the input of multiple sensors instead of a singular source. The sensors present include a Velodyne HDL-32E LiDAR, the Delphi Electronically Scanning RADAR (ESR) and consumer Logitech RGB cameras.

Detection networks broadly fall into two categories: two-stage and one-stage networks. Two-stage networks, like R-CNN [16] and its faster variants Fast R-CNN [17] and Faster R-CNN [18], first extract regions of objects and then, in the second stage, perform classification and further refine the localization of the object. One stage detectors, on the other hand, employ a single network to predict the bounding boxes and the corresponding class probabilities and usually exhibit higher inference speeds. Among one-stage detectors we can find Single Shot Detector (SSD) [19] and the You Only Look Once (YOLO) detector family [20]. In particular, a recent version of YOLOv4 [21] achieves real-time inference capabilities while maintaining a very high detection accuracy (Fig. 3). This network also comes in a version with a reduced number of layers YOLOv4 Tiny, that achieves a faster inference time at a cost of lower accuracy, specially suited for integration in embedded systems. More recently a new variant of YOLOv4, Scaled-YOLOv4 [22], can be scaled for both larger and smaller model sizes, providing techniques to automatically perform these scale changes. In this paper we will explore the smaller versions of Scaled-YOLOv4. As opposed to similar works like [23], [24], we test and evaluate the detector network real-time capabilities on a low power computational embedded system, that will be used for obstacle avoidance onboard the ORCA USV.

## III. Method

Training a deep neural detection network requires the use of massive amounts of data, due to the millions of parameters to be determined. Many detection networks are available in pre-trained versions, with parameters optimized for the detection of a certain number of predefined classes, which bring some major advantages: on one hand, this network can be immediately used to detect objects corresponding to classes for which the network has been pre-trained; on the other hand, even if the object class does not exist in such pre-trained version, it is generally much faster to adjust the network weights to this new class of objects, which requires a much smaller set of training data.

In this section we first introduce the datasets used to train the network; after that we briefly describe how the network was trained, and after that we provide some details on the implementation of the detection network on the embedded processing unit onboard the ORCA USV.

### A. Datasets

There are many maritime datasets for object detection: due to their publicly availability nature and the type of annotated classes, in these work we used the SeaShips [25] and ABOShips [23] datasets to train our detection network. The SeaShips dataset contains 31 455 images taken from video segments acquired from coastline video surveillance systems. All images are annotated with ship-type labels and high-precision bounding boxes corresponding to six classes: ore carrier, bulk cargo carrier, general cargo ship, container ship, fishing boat, and passenger ship. An example image from the SeaShips dataset is depicted in Fig. 4.

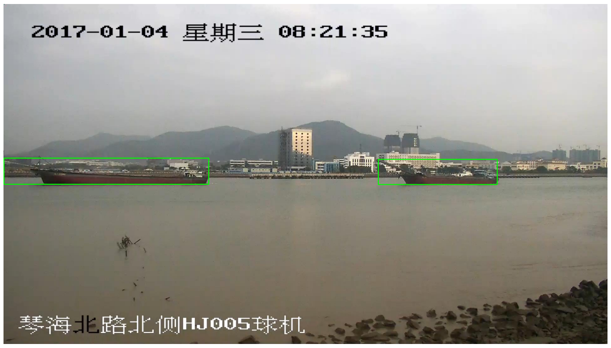

Fig. 4. Example annotated image from the Seaships dataset [25].

ABOShips dataset images, on the other hand, were obtained through a camera mounted on a ferry, providing footage from the point of view of the vehicle and providing annotations for nine types of vessels, seamarks and miscellaneous floaters. It

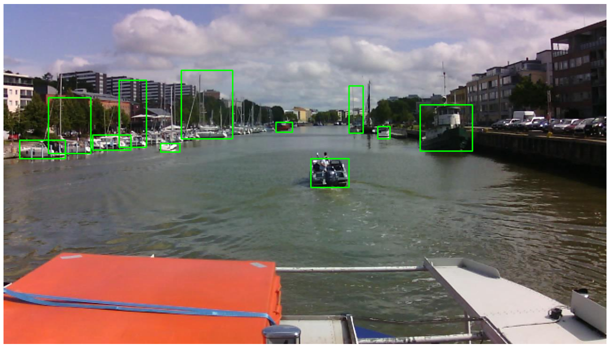

Fig. 5. Example annotated image from ABOShips dataset [23].

TABLE I
SUMMARY TABLE OF THE DIFFERENT CLASSES USED IN THIS WORK.

| Class | # Annotations | # Images |
|---|---|---|
| Motorized Vessel | 35 365 | 14 615 |
| Sailboat | 7670 | 3744 |
| Buoy/Seamark | 7670 | 3744 |

contains 9880 images containing a total of 41 967 annotations. A sample image from this dataset is presented in Fig. 5.

The SeaShips dataset has a clear representation of marine vessels, but does not represent sailboats or buoys which are essential classes for the proposed goal, which motivated the use of the ABOShips dataset. To train the network we combined images from these datasets. Due to the heterogeneity of classes between these two datasets we changed the annotations to take only into account three different classes: motorized vessel, sailboat and buoy/seamark. The number of images and annotations for the aggregated dataset are summarized in Table I.

### B. Training

Two versions of the YOLOv4 Tiny exist in the repository provided by the authors[7] [26]. While the first one (YOLOV4-tiny) has a 38-layer structure that allows for real-time inference on resource limited embedded computers like the Jetson Nano board, the second one (YOLOV4-tiny-3l) adds a detection layer that takes as input a finer grained feature map, to help improve the detection of smaller objects, at a cost of higher inference times. We will train, test and compare both models with respect to detection performance and inference time.

For YOLOV4-tiny the recommended size input is $416 \times 416$, while for YOLOV4-tiny-3l the recommended size input is $608 \times 608$. Since reducing resolution may decrease detection accuracy but increase inference speed, we will compare a wider variety of input resolutions, training both networks in five different configurations, corresponding to images with size $352 \times 352$, $416 \times 416$, $480 \times 480$, $544 \times 544$ and $608 \times 608$. We divide the dataset into a training set (70% of the data) and a validation set (remaining 30%): as usual, the first one is used to adjust the network parameters, while the validation images are used to calculate accuracy scores and decide on early stopping points to control overfit.

### C. Hardware Implementation

YOLO v4 detection network can be trained using high performance GPUs, a process that typically can take around 6 to 12 hours, taken from our own training runs using a NVIDIA GTX 1070. The obtained network weights must then be uploaded to the processing unit onboard the USV to enable the real-time detection of obstacles during the autonomous operation of the vehicle.

ORCA USV has a Linux based Jetson Nano board that is able to run the aforementioned trained models. To detect obstacles as fast as possible in this kind of embedded processing units a software library exists that allows for object detection models to be converted from their base frameworks into an optimized form, designed to run specifically on the target system. This optimized form of a neural network is called an *engine* and is created using NVIDIA's TensorRT[8]package. The optimizations made take several forms, of most note, TensorRT takes into account the architecture of the GPU and writes specific instructions on how the model should run calculations on it. These instructions are automatically generated and compiled into the *engine* by the software. The conversion process was made using the TensorRT_Demos repository[9], which provides a simple framework for converting YOLO networks into TensorRT engines.

## IV. RESULTS

To assess the detection capabilities of the trained YOLO v4 model we use the standard mean Average Precision (mAP) score, the area under the Precision-Recall curve as the detection threshold is varied, averaged for all the existing classes, and where Precision and Recall have the standard definitions:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \quad \text{and} \quad \text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \,,$$

with TP, FP and FN correspond to the number of detections, in the test set, categorized as true positives, false positives and false negatives, respectively. We consider a network detection to be a true positive if the Intersection Over Union (IoU) value between the predicted and ground-truth bounding boxes is above a given threshold for which the mAP value is obtained (*e.g.*, a mAP@0.5 is a mAP score obtained for detections corresponding to IoU $> 0.5$. Each YOLO network detection in the image comes with a confidence value (between 0 and 1), and a network detection is only considered if that confidence level is above a given detection threshold. By increasing the value of this threshold, there is more selectivity in what is considered a detection: this generally leads to an increase in precision and a corresponding decrease in recall.

Both YOLOv4-tiny and YOLOv4-tiny-3l were trained with multiple image resolutions on the SeaShips, ABOShips and

[7] https://github.com/AlexeyAB/darknet

[8] https://github.com/NVIDIA/TensorRT

[9] https://github.com/jkjung-avt/tensorrt_demos

TABLE II
DETECTION ACCURACY FOR YOLO V4 NETWORK AT DIFFERENT RESOLUTIONS AND TRAINED WITH DIFFERENT DATASETS.

| Model | Resolution | mAP@0.5 | | |
|---|---|---|---|---|
| | | SeaShips | ABOShips | Combined |
| yolov4-tiny | 352 × 352 | 84.37 | 35.24 | 49.77 |
| | 416 × 416 | 82.63 | 38.33 | 55.10 |
| | 480 × 480 | 85.63 | 40.02 | 56.88 |
| | 544 × 544 | 85.31 | 42.07 | 58.81 |
| | 608 × 608 | 84.63 | 42.89 | 60.28 |
| yolov4-tiny-3l | 352 × 352 | 84.06 | 37.39 | 54.86 |
| | 416 × 416 | 85.21 | 40.67 | 58.15 |
| | 480 × 480 | 83.98 | 41.54 | 59.64 |
| | 544 × 544 | 84.58 | 41.30 | 61.80 |
| | 608 × 608 | 83.45 | 42.88 | 62.09 |

combined datasets. The detection accuracy at mAP@0.5 is given in Table II.

In the SeaShips dataset, vessels are represented as large bounding boxes occupying most of the screen: as a result, input resolution has a reduced effect in detection accuracy, and the effect of the extra detection layer of the yolov4-tiny-3l model is not noticeable in the detection performance (Fig. 6).

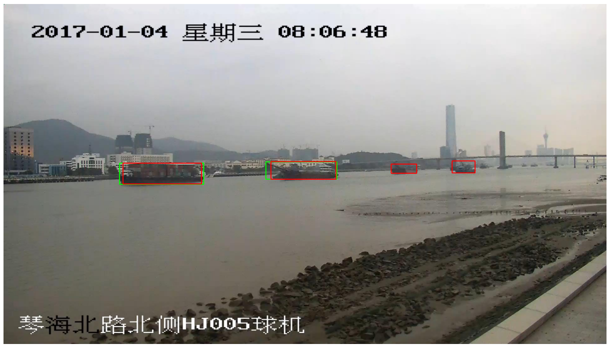

Fig. 6. Example of detections in the Seaships dataset [25]. In green, annotations, in red, detections.

On the other hand, the detection on the ABOShips dataset achieves a lower value of mAP@0.5, as a result of image objects occupying smaller portions of the image when compared to the SeaShips dataset. As expected, training and evaluating the performance on the combined dataset with 3 different classes achieves an intermediate result with respect to the mAP@0.5. In this latter dataset a performance gain when using the YOLOv4-tiny-3l model is more noticeable (Fig. 7).

To evaluate detection speed we employ the same sample video in all tests. We present inference times, measured in frames per second (fps) for the models trained on the combined dataset, corresponding to the average fps value for the duration of the video sample. We run these tests on the Jetson Nano using both the Darknet implementation and the model engine obtained through TensorRT, obtaining the results presented in Table III.

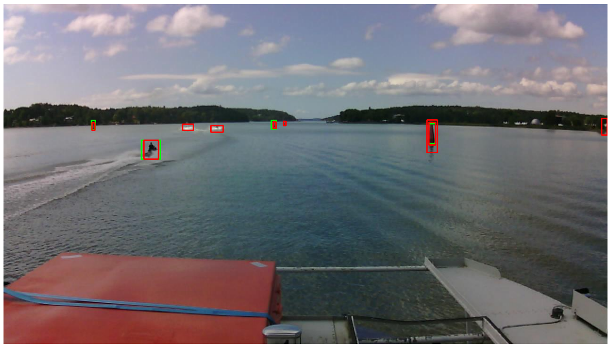

Fig. 7. Example of detections in the ABOShips dataset [23]. In green, annotations, in red, detections.

TABLE III
INFERENCE SPEED COMPARISON: YOLO V4 RUNNING ON JETSON NANO.

| Model | Resolution | Inference (fps) | |
|---|---|---|---|
| | | Darknet | TensorRT |
| yolov4-tiny | 352 × 352 | 20.6 | 30.5 |
| | 416 × 416 | 16.1 | 24.8 |
| | 480 × 480 | 13.8 | 21.6 |
| | 544 × 544 | 9.3 | 15.7 |
| | 608 × 608 | 8.3 | 14.3 |
| yolov4-tiny-3l | 352 × 352 | 18.7 | 27.9 |
| | 416 × 416 | 14.4 | 22.9 |
| | 480 × 480 | 12.3 | 19.3 |
| | 544 × 544 | 8.4 | 14.5 |
| | 608 × 608 | 7.5 | 12.9 |

This table show us that we achieve a performance increase of around 50% when using the TensorRT optimized version of the Yolo v4 Tiny network. This decrease in inference time, however, does not correspond to any noticeable detection accuracy loss, as can be observed in Table IV.

TABLE IV
DETECTION ACCURACY MAP@0.5 COMPARISON: YOLO V4 RUNNING ON JETSON NANO.

| Model | Resolution | mAP@0.5 | |
|---|---|---|---|
| | | Darknet | TensorRT |
| yolov4-tiny | 352 × 352 | 50.08 | 49.27 |
| | 416 × 416 | 55.10 | 54.21 |
| | 480 × 480 | 57.14 | 56.94 |
| | 544 × 544 | 59.06 | 58.38 |
| | 608 × 608 | 60.28 | 59.75 |
| yolov4-tiny-3l | 352 × 352 | 54.86 | 53.92 |
| | 416 × 416 | 58.36 | 57.29 |
| | 480 × 480 | 60.02 | 59.56 |
| | 544 × 544 | 61.91 | 60.72 |
| | 608 × 608 | 62.23 | 61.20 |

For visualization purpose, a small video sequence, taken from the ORCA USV, is available on YouTube, depicting the detection bounding boxes provided by the trained network, specifically, yolov4-tiny-3l with 416x416 input resolution[10].

## V. Discussion and Future Work

The recent developments in detection using deep neural networks allow the integration of state-of-the-art detectors onboard an USV without the need to use costly and dedicated hardware. We show in this work that the YOLO v4 can be successfully trained on publicly available datasets to detect maritime obstacles. As opposed to LiDAR and radar approaches, the proposed method is able to differentiate between different types of obstacles and thus makes possible the enforcement of maritime collision regulations. Additionally, we show that such detection network exhibits a real-time performance when TensorRT is used, surpassing the 10 fps mark even at higher image resolutions.

To fully integrate this detector in the navigation and control software running onboard some topics must be first addressed:

- A new dataset with images taken from the ORCA USV camera in different times of day and in different atmospheric conditions should be collected and annotated, to train the YOLO detection network with images corresponding to the situations where inference will occur;
- Inclusion of a tracker to ensure temporal coherence between detections in different time frames and outlier rejection, *e.g.*, by resorting to multiple object trackers like [27]–[29].
- Calculation of obstacles locations in vehicle/world coordinates. Conversion from image coordinates can be implemented resorting to the knowledge of camera intrinsic and extrinsic parameters, but obtaining depth information must resort to the use of two cameras [30] or to data fusion with other sensors like LiDAR or radar [15].

These topics will be addressed in future work.

## References

[1] R.-j. Yan, S. Pang, H.-b. Sun, and Y.-j. Pang, "Development and missions of unmanned surface vehicle," *Journal of Marine Science and Application*, vol. 9, 2010.

[2] J. E. Manley, "Development of the autonomous surface craft" aces"," in *Oceans' 97. MTS/IEEE Conference Proceedings*, vol. 2. IEEE, 1997, pp. 827–832.

[3] J. Curcio, J. Leonard, and A. Patrikalakis, "Scout-a low cost autonomous surface platform for research in cooperative autonomy," in *Proceedings of OCEANS 2005 MTS/IEEE*. IEEE, 2005, pp. 725–729.

[4] A. Pascoal, P. Oliveira, C. Silvestre, L. Sebastião, M. Rufino, V. Barroso, J. Gomes, G. Ayela, P. Coince, M. Cardew *et al.*, "Robotic ocean vehicles for marine science applications: the european asimov project," in *OCEANS 2000 MTS/IEEE Conference and Exhibition. Conference Proceedings (Cat. No. 00CH37158)*, vol. 1. IEEE, 2000, pp. 409–415.

[5] A. Martins, J. Almeida, E. Silva, and F. Pereira, "Vision-based autonomous surface vehicle docking manoeuvre," in *Proc. of 7th IFAC conference on manoeuvring and control of marine craft*, 2006.

[6] M. Caccia, M. Bibuli, R. Bono, and G. Bruzzone, "Basic navigation, guidance and control of an unmanned surface vehicle," *Autonomous Robots*, vol. 25, no. 4, pp. 349–365, 2008.

[7] P. Tang, R. Zhang, D. Liu, L. Huang, G. Liu, and T. Deng, "Local reactive obstacle avoidance approach for high-speed unmanned surface vehicle," *Ocean engineering*, vol. 106, pp. 128–140, 2015.

[8] R. Polvara, S. Sharma, J. Wan, A. Manning, and R. Sutton, "Obstacle avoidance approaches for autonomous navigation of unmanned surface vehicles," *The Journal of Navigation*, vol. 71, no. 1, pp. 241–256, 2018.

[9] Y. Wang, X. Yu, X. Liang, and B. Li, "A colregs-based obstacle avoidance approach for unmanned surface vehicles," *Ocean Engineering*, vol. 169, pp. 110–124, 2018.

[10] Y. Kuwata, M. T. Wolf, D. Zarzhitsky, and T. L. Huntsberger, "Safe maritime autonomous navigation with colregs, using velocity obstacles," *IEEE Journal of Oceanic Engineering*, vol. 39, no. 1, pp. 110–119, 2013.

[11] B. Bovcon and M. Kristan, "Wasr – a water segmentation and refinement maritime obstacle detection network," *IEEE Transactions on Cybernetics*, pp. 1–14, 2021.

[12] ——, "Obstacle detection for usvs by joint stereo-view semantic segmentation," IEEE, Oct 2018.

[13] B. Bovcon, R. Mandeljc, J. Pers, and M. Kristan, "Stereo obstacle detection for unmanned surface vehicles by imu-assisted semantic segmentation," *CoRR*, vol. abs/1802.07956, 2018. [Online]. Available: http://arxiv.org/abs/1802.07956

[14] X. Chen, Y. Liu, and K. Achuthan, "Wodis: Water obstacle detection network based on image segmentation for autonomous surface vehicles in maritime environments," *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1–13, 2021.

[15] L. Stanislas and M. Dunbabin, "Multimodal sensor fusion for robust obstacle detection and classification in the maritime robotx challenge," *IEEE Journal of Oceanic Engineering*, vol. 44, no. 2, pp. 343–351, 2019.

[16] R. B. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," *CoRR*, vol. abs/1311.2524, 2013. [Online]. Available: http://arxiv.org/abs/1311.2524

[17] R. B. Girshick, "Fast R-CNN," *CoRR*, vol. abs/1504.08083, 2015. [Online]. Available: http://arxiv.org/abs/1504.08083

[18] S. Ren, K. He, R. B. Girshick, and J. Sun, "Faster R-CNN: towards real-time object detection with region proposal networks," *CoRR*, vol. abs/1506.01497, 2015. [Online]. Available: http://arxiv.org/abs/1506.01497

[19] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. E. Reed, C. Fu, and A. C. Berg, "SSD: single shot multibox detector," *CoRR*, vol. abs/1512.02325, 2015. [Online]. Available: http://arxiv.org/abs/1512.02325

[20] J. Redmon, S. K. Divvala, R. B. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," *CoRR*, vol. abs/1506.02640, 2015. [Online]. Available: http://arxiv.org/abs/1506.02640

[21] A. Bochkovskiy, C. Wang, and H. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," *CoRR*, vol. abs/2004.10934, 2020. [Online]. Available: https://arxiv.org/abs/2004.10934

[22] C. Wang, A. Bochkovskiy, and H. M. Liao, "Scaled-yolov4: Scaling cross stage partial network," *CoRR*, vol. abs/2011.08036, 2020. [Online]. Available: https://arxiv.org/abs/2011.08036

[23] B. Iancu, V. Soloviev, L. Zelioli, and J. Lilius, "Aboships—an inshore and offshore maritime vessel detection dataset with precise annotations," *Remote Sensing*, vol. 13, no. 5, 2021. [Online]. Available: https://www.mdpi.com/2072-4292/13/5/988

[24] T. Liu, B. Pang, S. Ai, and X. Sun, "Study on visual detection algorithm of sea surface targets based on improved yolov3," *Sensors*, vol. 20, no. 24, p. 7263, 2020.

[25] Z. Shao, W. Wu, Z. Wang, W. Du, and C. Li, "Seaships: A large-scale precisely annotated dataset for ship detection," *IEEE Transactions on Multimedia*, vol. 20, no. 10, pp. 2593–2604, 2018.

[26] J. Redmon, "Darknet: Open source neural networks in C," http://pjreddie.com/darknet/, 2013–2016.

[27] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, "Simple online and realtime tracking," *CoRR*, vol. abs/1602.00763, 2016. [Online]. Available: http://arxiv.org/abs/1602.00763

[28] Y. Zhang, P. Sun, Y. Jiang, D. Yu, Z. Yuan, P. Luo, W. Liu, and X. Wang, "Bytetrack: Multi-object tracking by associating every detection box," *arXiv preprint arXiv:2110.06864*, 2021.

[29] Y. Zhang, C. Wang, X. Wang, W. Zeng, and W. Liu, "Fairmot: On the fairness of detection and re-identification in multiple object tracking," *International Journal of Computer Vision*, vol. 129, no. 11, pp. 3069–3087, 2021.

[30] N. Smolyanskiy, A. Kamenev, and S. Birchfield, "On the importance of stereo for accurate depth estimation: An efficient semi-supervised deep neural network approach," 06 2018, pp. 1120–11 208.

[10] https://youtu.be/4B8iUF5lD3s